ELSEVIER

Discrete Applied Mathematics 116 (2002) 243–260

DISCRETE APPLIED MATHEMATICS

Partitioning a matrix with non-guillotine cuts to minimize the maximum cost

Aristide Mingozzi *, Serena Morigi

Department of Mathematics, Scienze dell-Informazione, University of Bologna, Piazza di Porta San Donato 5, 40126 Bologna, Italy

Received 30 January 1996; revised 7 June 2000; accepted 19 June 2000

Abstract

We consider the problem of partitioning a matrix of m rows and n columns of non-negative integers into M smaller submatrices. With each submatrix is associated a cost equal to the sum of its elements. The objective is to minimize the cost of the submatrix of maximum cost. We present a (0–1)-integer programming formulation of the problem and three different lower bounds. A heuristic procedure for finding a valid upper bound to the optimal solution cost is also described. Problem reduction tests derived from both the original problem and the lower bounds are given. Lower bounds and reduction tests are used in a tree search algorithm in order to find the optimal solution to the problem. Computational results on a number of randomly generated test problems are presented.

Keywords: Matrix partitioning; Integer programming; Combinatorial optimization; Branch and bound techniques

1. Introduction

In the matrix partitioning problem (MP) a matrix $A=[a_{ij}]$ of m rows and n columns of non-negative integers is to be partitioned into M smaller submatrices of different sizes. Let $A_{ij\alpha\beta}$ be a submatrix that contains all elements a_{rs} such that $i\leqslant r\leqslant j$ and $\alpha\leqslant s\leqslant\beta$. With each submatrix $A_{ij\alpha\beta}$ we associate a cost $c_{ij\alpha\beta}=\sum_{r=i}^{j}\sum_{s=\alpha}^{\beta}a_{rs}$ that corresponds to the sum of its elements. The objective is to find a matrix partitioning that minimizes the cost of the submatrix of maximum cost.

A similar problem, with the additional constraint that the submatrices must be obtained by means of a set of vertical and horizontal cuts (guillotine cuts), has been

* Corresponding author.
E-mail addresses: mingozzi@dm.unibo.it (A. Mingozzi), morigi@dm.unibo.it (S. Morigi).

0166-218X/02/$ - see front matter
PII: S0166-218X(00)00286-9

0	4	0	0	72	0	87	77	0	0
0	4	34	77	0	17	69	94	0	0
54	2	0	0	90	104	87	1	104	0
8	92	25	23	0	52	0	0	87	10
11	3	5	45	78	61	67	27	2	88
35	63	103	102	59	4	0	8	11	65
3	5	26	0	0	0	8	19	9	14
75	5	2	5	2	26	0	0	21	0
26	8	60	0	0	0	1	0	35	0
60	79	13	9	44	0	1	0	95	0

Fig. 1. Optimal partitioning of a matrix A (10×10) into $M = 12$ submatrices using guillotine cuts.

described by Mingozzi et al. [8]. The problem considered in this paper is more general in that the partitioning is not restricted to guillotine cuts.

Becker et al. [3] studied a somewhat different max–min partitioning problem where a rectangular grid graph with weighted vertices must be partitioned into M connected components. This problem becomes the MP studied in this paper when each component is restricted to be a rectangular subgraph.

To our knowledge no exact algorithm exists in the literature for solving problem MP with non-guillotine cuts.

In Fig. 1, an optimal partitioning of a matrix $A(10\times10)$ into $M=12$ submatrices using guillotine cuts is shown. The partitioning has a maximum cost of 349 corresponding to the cost of submatrix A_{5746}.

In Fig. 2, an optimal partitioning of the matrix A given in Fig. 1 into 12 submatrices using non-guillotine cuts is shown. Here we found a partitioning of A of cost of 257 corresponding to the cost of submatrix A_{2248}.

An application of problem MP arises in the management of a parallel computer system having M processors where a problem workload, represented by matrix A, must be partitioned and distributed to the processors (see [5,6,9,10,12]). In order to maximize the system performance the workload must be partitioned in an evenly balanced way. In a wide range of applications, including computer graphics, image processing, and numerical analysis, the workload corresponds to a matrix. Examples of workloads, that can be easily represented by a matrix, range from the projection of a 3D scene, created by computer graphics techniques, to large matrices involved in numerical computations and images resulting from image processing algorithms. The value of each matrix element represents the number of computations required. For example, in a matrix

0	4	0	0	72	0	87	77	0	0
0	4	34	77	0	17	69	94	0	0
54	2	0	0	90	104	87	1	104	0
8	92	25	23	0	52	0	0	87	10
11	3	5	45	78	61	67	27	2	88
35	63	103	102	59	4	0	8	11	65
3	5	26	0	0	0	8	19	9	14
75	5	2	5	2	26	0	0	21	0
26	8	60	0	0	0	1	0	35	0
60	79	13	9	44	0	1	0	95	0

Fig. 2. Optimal partitioning of the matrix A of Fig. 1 into 12 submatrices by means of non-guillotine cuts.

of pixels representing an image, the value of each matrix element can represent the number of ray/surface intersections that a ray tracing algorithm must compute for the corresponding pixel.

The workload of each processor depends on the sum of the values of the submatrix elements assigned to it. This problem can be tackled in two ways: either by allocating individual matrix elements to the processors, or by first partitioning the matrix into M contiguous subregions and subsequently allocating each subregion to a different processor. This latter approach (see [6]) is preferred in those applications (e.g. image processing and computer graphics) where it must be considered the overhead associated with inter-processor communications. This overhead is proportional to the number of adjacent pairs of matrix elements assigned to different processors. The communication costs can be strongly reduced forcing the subregions to be of rectangular shape (see [9]). In this case the workload balancing problem can be formulated as the MP problem considered in this paper. At our knowledge, the methods proposed in the literature for this problem only consider solutions obtained by orthogonal recursive bisections (see [11]).

Two applications related to MP are described by Becker et al. [3]. In the first application matrix A represents the individual units of a manufacturing plant that must be partitioned among M supervisors such that the workload of the supervisors is as "balanced" as possible. The second application deals with the balanced subdivision of a rectangular mining area among M mining companies.

In this paper we present a (0–1) integer programming formulation of problem MP and three different lower bounds that are used in a tree search algorithm for finding the optimal solution. Problem reduction tests derived from both the original problem

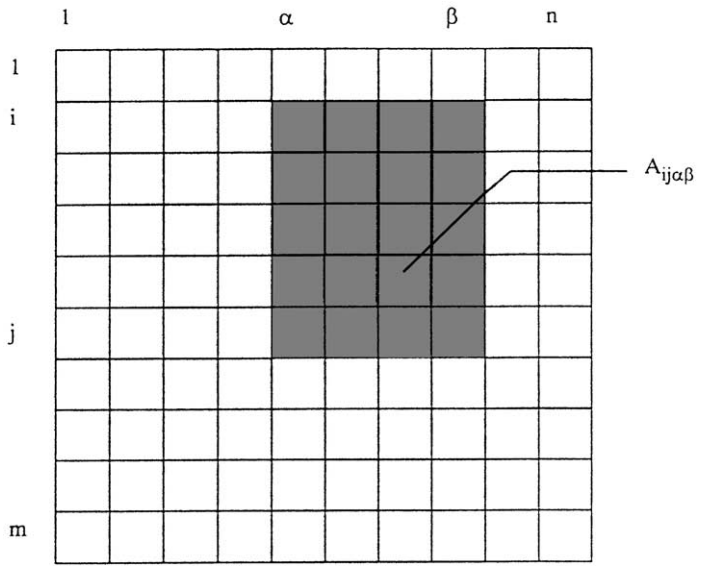

Fig. 3. Locating a submatrix $A_{ij\alpha\beta}$.

and the lower bounds are given. A heuristic procedure for finding a valid upper bound is also described.

The computational results show that moderately sized problems can be solved by the proposed algorithm.

2. Mathematical formulation

In this section we describe a (0–1)-integer programming formulation of problem MP. The formulation is based on one proposed by Beasley for the two-dimensional non-guillotine cutting problem (see [2]) and will be used in Section 3 to derive valid lower bounds that can be incorporated into an exact tree-search procedure.

2.1. Formulation MP

Let S be the index set of all possible submatrices of A, where each $\ell \in S$ denotes the submatrix $A_{i_\ell j_\ell \alpha_\ell \beta_\ell}$ (see Fig. 3). Let $S_{rs} \subset S$ be the index subset of all the submatrices covering the matrix element a_{rs}. Hereafter, we will use the shortening c_ℓ to denote the cost $c_{i_\ell j_\ell \alpha_\ell \beta_\ell}$ of the submatrix of index ℓ, and R_ℓ to represent the set of row and column indices of the elements of matrix A covered by submatrix ℓ, that is

$$R_\ell = \{(r,s)\colon i_\ell \leqslant r \leqslant j_\ell \quad \text{and} \quad \alpha_\ell \leqslant s \leqslant \beta_\ell\}.$$

Let x_ℓ be a (0–1) binary variable that is equal to 1 if and only if the submatrix of index ℓ is in the optimal solution.

The mathematical formulation of problem MP is

$$\text{(MP):} \quad \text{Min} \quad z \tag{1}$$

$$\text{s.t.} \quad z \geqslant c_\ell x_\ell, \quad \ell \in S, \tag{2}$$

$$\sum_{\ell \in S_{rs}} x_\ell = 1, \quad r = 1, \ldots, m, \;\; s = 1, \ldots, n, \tag{3}$$

$$\sum_{\ell \in S} x_\ell = M, \tag{4}$$

$$x_\ell \in \{0, 1\}, \quad \ell \in S. \tag{5}$$

Constraint (2) ensures that the cost of the solution is greater than or equal to the cost of the submatrix of maximum cost. Eq. (3) ensure that every matrix element a_{rs} is covered exactly once, while Eq. (4) forces the solution of contain M submatrices.

Formulation MP requires $\hat{n} = n(n+1)m(m+1)/4$ variables and $\hat{n} + mn + 1$ constraints. However, the number of variables (and consequently the number of constraints of type (2)) can be reduced by means of the reduction tests described in the following section.

2.1.1. Variable reduction

The set S can be reduced by means of the following observations.

Reduction 1. Let z_{UB} be a valid upper bound to the optimal solution cost found by a heuristic algorithm. It is obvious that any optimal solution of cost $z^* < z_{\mathrm{UB}}$ cannot contain a submatrix $\ell \in S$ of cost $c_\ell \geqslant z_{\mathrm{UB}}$, hence every such submatrix can be removed from S.

Reduction 2. Any optimal solution of cost $z^* < z_{\mathrm{UB}}$ cannot contain any submatrix $\ell \in S$ of cost $c_\ell < \hat{c}$, where

$$\hat{c} = \sum_{r=1}^{m} \sum_{s=1}^{n} a_{rs} - (M-1)(z_{\mathrm{UB}} - 1). \tag{6}$$

Proof. By contradiction. Assume that there exists an optimal solution of cost $z^* < z_{\mathrm{UB}}$ containing a submatrix $\ell^* \in S$ of cost $c_{\ell^*} < \hat{c}$. The cost of any other submatrix of such solution must be greater than or equal to $\bar{c}$ where

$$\bar{c} = \frac{1}{(M-1)} \left(\sum_{r=1}^{m} \sum_{s=1}^{n} a_{rs} - c_{\ell^*} \right).$$

Since $c_{\ell^*} < \hat{c}$, we have

$$\bar{c} > \frac{1}{(M-1)} \left(\sum_{r=1}^{m} \sum_{s=1}^{n} a_{rs} - \hat{c} \right). \tag{7}$$

Substituting $\hat{c}$ into Eq. (7), we obtain

$$\bar{c} > \frac{1}{(M-1)}\left(\sum_{r=1}^{m}\sum_{s=1}^{n} a_{rs} - \left(\sum_{r=1}^{m}\sum_{s=1}^{n} a_{rs} - (M-1)(z_{\mathrm{UB}}-1)\right)\right)$$

or $\bar{c} > z_{\mathrm{UB}} - 1$, hence, $z^* \geqslant \bar{c} > z_{\mathrm{UB}} - 1$, giving a contradiction. □

In the following, we will assume that the set S is such that

$$\hat{c} \leqslant c_\ell < z_{\mathrm{UB}}, \quad \forall \ell \in S. \tag{8}$$

2.1.2. Linear programming relaxation of MP

A valid lower bound to the optimal MP solution can be obtained from MP by relaxing the integrality constraints (5) to $x_\ell \geqslant 0$, $\forall \ell \in S$ and solving resulting problem, called LP.

However, this bound can be very weak, as shown in the following example. Let us consider the following matrix

$$A = \begin{bmatrix} 0 & 0 \\ L & 0 \end{bmatrix}$$

where L is a positive integer, to be partitioned into $M = 2$ submatrices. The list of all possible submatrices of A is $(A_{1111}, A_{1112}, A_{1211}, A_{1212}, A_{1122}, A_{1222}, A_{2211}, A_{2212}, A_{2222})$ and the cost vector of these submatrices is $c = (0,0,L,L,0,0,L,L,0)$. We assume that the variable x_1 represents the first submatrix A_{1111}, x_2 represents the second submatrix A_{1112}, and so on.

An optimal solution of problem LP of cost $z_{\mathrm{LP}} = L/4$ is $x^*_{\mathrm{LP}} = (\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}, 0, \frac{1}{2}, \frac{1}{4}, \frac{1}{4}, 0)$ while the cost of any optimal solution of MP2 is $z = \mathrm{L}$. Other more complex test problems have confirmed that the gap between the optimal solution cost of LP and the optimal MP solution cost is too large and, therefore, this bound has not been investigated further.

3. A method for computing lower bounds

An obvious lower bound, called LB0, to the optimal solution cost z^* can be computed as follows:

$$\mathrm{LB0} = \max\left\{\left\lceil \frac{1}{M}\sum_{r=1}^{m}\sum_{s=1}^{n} a_{rs} \right\rceil, \max_{r,s}[a_{rs}]\right\}, \tag{9}$$

where $\lceil x \rceil$ denotes the smallest integer greater than or equal to x.

In this section we describe a method that is used in Sections 4 and 5 for computing two new valid lower bounds LB1 and LB2 such that LB1 $\geqslant$ LB0 and LB2 $\geqslant$ LB0. The method for computing LB1 and LB2 is based on the observation that any optimal solution of cost $z^* = q$ is a feasible solution to the problem of partitioning matrix A into M submatrices where each submatrix has a cost smaller than or equal to q. This latter

problem, called MP(q), can be formulated as follows. Let $w_\ell = (\beta_\ell - \alpha_\ell + 1)(j_\ell - i_\ell + 1)$ be the number of elements (area) of the submatrix $\ell \in S$, and $S(q) = \{\ell: \ell \in S \text{ and } c_\ell \leqslant q\}$, be the index subset of all submatrices of cost less than q, and $S_{rs}(q) = S_{rs} \cap S(q)$, be the index subset of all submatrices of cost less than q covering the matrix element a_{rs}.

Problem (MP(q)) is as follows:

$$(\mathrm{MP}(q)): \quad \sigma(q) = \mathrm{Max} \sum_{\ell \in S(q)} w_\ell x_\ell \tag{10}$$

$$\text{s.t.} \quad \sum_{\ell \in S_{rs}(q)} x_\ell = 1, \quad r = 1, \ldots, m, \ \ s = 1, \ldots, n, \tag{11}$$

$$\sum_{\ell \in S(q)} x_\ell = M, \tag{12}$$

$$x_\ell \in \{0, 1\}, \quad \ell \in S(q). \tag{13}$$

It is obvious that if MP(q) contains a solution of cost $\sigma(q) = mn$, then problem MP has an optimal solution of cost $z^* \leqslant q$. Notice that the cost of any feasible solution of MP(q) is equal to mn. We assume $\sigma(q) = -\infty$ if MP(q) has no feasible solution. Therefore, the optimal MP solution z^* can be obtained by finding z^* such that

$$z^* = \min\{q: \ \mathrm{LBO} \leqslant q < z_{\mathrm{UB}} \quad \textit{and} \quad \sigma(q) = mn\}. \tag{14}$$

A valid lower bound LB to z^* can be obtained from expression (14) as follows:

$$\mathrm{LB} = \min\{q: \ \mathrm{LBO} \leqslant q < z_{\mathrm{UB}} \text{ s.t } \mathrm{UB}(q) \geqslant mn\}, \tag{15}$$

where UB(q) is a valid upper bound to $\sigma(q)$ and can be obtained by finding the optimal solution of the relaxed problem RMP(q, λ) derived from MP(q) by relaxing, in a Lagrangean fashion, constraints (11).

Problem (RMP(q, λ)) is as follows:

$$(\mathrm{RMP}(q, \lambda)): \quad \sigma(q, \lambda) = \mathrm{Max} \sum_{\ell \in S(q)} w'_\ell x_\ell + \sum_{r=1}^{m} \sum_{c=1}^{n} \lambda_{rs}$$
$$\text{s.t.} \quad (12) \text{ and } (13),$$

where $w'_\ell = w_\ell - \sum_{r=\beta_\ell}^{\alpha_\ell} \sum_{c=j_\ell}^{i_\ell} \lambda_{rs}$, and the upper bound UB(q) is then given by

$$\mathrm{UB}(q) = \min_{\lambda} \{\sigma(q, \lambda)\}$$

and can be computed using subgradient optimization techniques (see [4]).

In the next Sections 4 and 5, we describe two lower bounds, called LB1 and LB2, respectively, that are obtained by adding to problem RMP(q, λ) two different types of constraints. These constraints are redundant with respect to Eq. (11) of problem MP(q); however, they are not redundant in RMP(q, λ), consequently, the resulting value of $\sigma(q, \lambda)$ can be smaller than UB(q), thus improving the value of LB given by expression (15).

4. Lower bound LB1

Let $P=\{(r_1,s_1),(r_2,s_2),\ldots,(r_p,s_p)\}$ be an ordered set of the indices of p elements of A, with $p\leqslant mn$, inducing a partitioning of S into p disjointed subsets:

$$S=S'_{r_1s_1}\cup S'_{r_2s_2}\cup\cdots\cup S'_{r_ps_p}, \tag{16}$$

where, $\forall(r_k,s_k)\in P$, we have

$$S'_{r_ks_k}\subseteq S_{r_ks_k}, \tag{17}$$

$$R_\ell\cap\{(r_1,s_1),(r_2,s_2),\ldots,(r_{k-1},s_{k-1})\}=\emptyset,\quad\forall\ell\in S'_{r_ks_k}. \tag{18}$$

We assume that P is partitioned as $P=P'\cup P''$, where P' corresponds to the first p' elements of P and P'' to the last $p''=p-p'$ elements of P. We impose that any two elements (r_i,s_i), (r_j,s_j) belonging to P', cannot be covered by any submatrix $\ell\in S$, that is

$$R_\ell\cap(P'\setminus\{(r_k,s_k)\})=\emptyset,\quad\forall\ell\in S'_{r_ks_k},\ \forall(r_k,s_k)\in P'. \tag{19}$$

From condition (19) we have that every submatrix of the set $\bigcup_{(r,s)\in P'}S'_{rs}$ covers exactly one element of P'. Notice that condition (18) implies that

$$S'_{r_ks_k}=S_{r_ks_k}\setminus\left(S_{r_ks_k}\cap\left(\bigcup_{i=1}^{k-1}S'_{r_is_i}\right)\right),\quad\forall(r_k,s_k)\in P \tag{20}$$

and from condition (19) we have $S'_{rs}=S_{rs}$, for each $(r,s)\in P'$.

For a given ordered set P satisfying conditions (18) and (19) the following constraints can be added to RMP(q,λ):

$$\sum_{\ell\in S'_{rs}(q)}x_\ell\begin{cases}=1 & \text{if } (r,s)\in P',\\ \leqslant 1 & \text{if } (r,s)\in P''.\end{cases} \tag{21}$$

We can observe that the quality of the upper bound UB(q) may be strongly affected by the partition P used in generating constraints (21). A method for computing the sets P' and P'' is described in [7].

4.1. Computation of lower bound LB1

We call LB1 the value of the lower bound computed according to expression (15), where $\mathrm{UB}(q)=\min_\lambda[\sigma'(q,\lambda)]$ and $\sigma'(q,\lambda)$ is the optimal solution cost of the following problem RMP1(q,λ) that is obtained from problem RMP(q,λ) by adding constraints (21), that is

$$(\mathrm{RMP1}(q,\lambda))\colon\ \sigma'(q,\lambda)=\mathrm{Max}\sum_{\ell\in S(q)}w'_\ell x_\ell+\sum_{r=1}^{m}\sum_{s=1}^{n}\lambda_{rs} \tag{22}$$

$$\text{s.t.}\sum_{\ell\in S(q)}x_\ell=M, \tag{23}$$

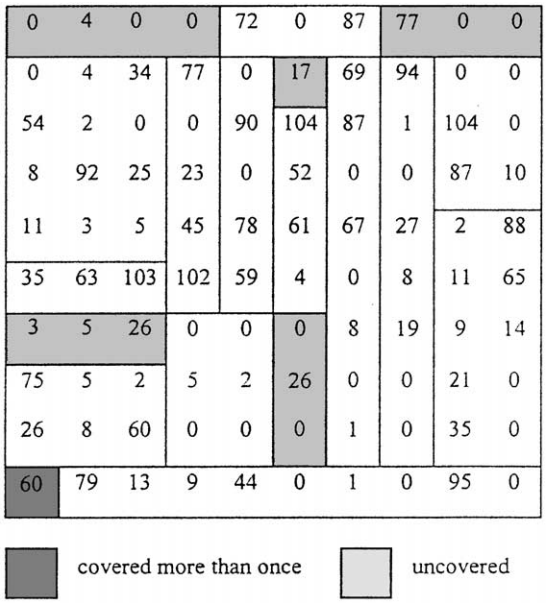

Fig. 4. Example of lower bound LB1 for the problem of Fig. 1.

$$\sum_{\ell \in S'_{rs}(q)} x_\ell \begin{cases} =1 & \text{if } (r,s) \in P', \\ \leqslant 1 & \text{if } (r,s) \in P'', \end{cases} \tag{24}$$

$$x_\ell \in \{0,1\}, \quad \ell \in S(q). \tag{25}$$

An optimal solution of RMP1(q,λ) can be easily computed as follows:

(i) Let $h_{rs} = \max_{\ell \in S'_{rs}(q)}[w'_\ell]$; we assume $h_{rs} = 0$ if $S'_{rs}(q) = \emptyset$.
(ii) Sort P'' in non-increasing h_{rs} order: $P'' = ((r_{i_1}, s_{i_1}), (r_{i_2}, s_{i_2}), \ldots, (r_{i_t}, s_{i_t}))$ where $h_{r_{i_1} s_{i_1}} \geqslant h_{r_{i_2} s_{i_2}} \geqslant \cdots \geqslant h_{r_{i_t} s_{i_t}}$.
(iii) Then $\sigma'(q,\lambda) = \sum_{(r,s)\in P'} h_{rs} + \sum_{t=1}^{M-|P'|} h_{r_{i_t} s_{i_t}}$.

For the MP problem shown in Fig. 1, the value of LB1 is 247, which corresponds to the solution shown in Fig. 4.

5. Lower bound LB2

Lower bound LB2 is derived from LB1 imposing the additional constraint that every border element of matrix A is covered exactly by one submatrix.

Let $B = ((r_1, s_1), (r_2, s_2), \ldots, (r_{|B|}, s_{|B|}))$ be an ordered set containing the $(2m+2n-4)$ indices of the border elements of matrix A. We assume that B is ordered as follows: $B = ((1,1), \ldots, (1,n), (2,n), \ldots, (m,n), (m,n-1), \ldots, (m,1), (m-1,1), \ldots, (2,1))$.

Notice that if the number of required partitions M is greater than or equal to 4, then any feasible solution must contain at least 4 submatrices covering B and at most $(M-4)$ submatrices not covering any element of B.

The submatrices not covering elements of B can be partitioned by means of a subset P of elements of A, as described in Section 4, such that $P \cap B = \emptyset$.

We denote by RMP2(q,λ) the problem obtained from RMP(q,λ) by adding constraints (21) and the $(2m+2n-4)$ constraints of type (3) corresponding to the matrix elements in B.

$$(\text{RMP2}(q,\lambda)): \quad \sigma(q,\lambda) = \text{Max} \sum_{\ell \in S(q)} w'_\ell x_\ell + \sum_{r=1}^{m}\sum_{s=1}^{n} \lambda_{rs} \tag{26}$$

$$\text{s.t.} \sum_{\ell \in S'_{rs}(q)} x_\ell \begin{cases} =1, & (r,s) \in P', \\ \leqslant 1, & (r,s) \in P'', \end{cases} \tag{27}$$

$$\sum_{\ell \in S(q)} x_\ell = M, \tag{28}$$

$$\sum_{\ell \in S_{rs}(q)} x_\ell = 1, \quad (r,s) \in B, \tag{29}$$

$$x_\ell \in \{0,1\}, \quad \ell \in S(q). \tag{30}$$

Problem RMP2(q,λ) is as hard to solve as problem MP(q); however, it can be relaxed as follows. Let $\mathscr{B}(q) \subseteq S(q)$ be the subset of the submatrices covering at least one element of B and $\mathscr{B}'(q) = S(q) \setminus \mathscr{B}(q)$ and let $\mathscr{B}'_{rs}(q) = S'_{rs}(q) \cap \mathscr{B}'(q)$, $\forall (r,s) \in B$. Consider the relaxed problem RRMP2(q,λ) that is obtained from RMP2(q,λ) by replacing in constraints (27) "=" with "$\leqslant$" and the set $S'_{rs}(q)$ with $\mathscr{B}'_{rs}(q)$. Furthermore, we replace in constraints (29) the set $S_{rs}(q)$ with $\mathscr{B}_{rs}(q) = S_{rs}(q) \cap \mathscr{B}(q)$, $\forall (r,s) \notin B$.

Problem RRMP2(q,λ) can be written as follows:

$$(\text{RRMP2}(q,\lambda)): \quad \sigma''(q,\lambda) = \text{Max} \quad \sum_{l \in \mathscr{B}(q)} w'_\ell x_\ell + \sum_{\ell \in \mathscr{B}'(q)} w'_\ell x_\ell + \sum_{r=1}^{m}\sum_{s=1}^{n} \lambda_{rs}$$

$$\text{s.t.} \quad \sum_{\ell \in \mathscr{B}(q)} x_\ell + \sum_{\ell \in \mathscr{B}'(q)} x_\ell = M, \tag{31}$$

$$\sum_{\ell \in \mathscr{B}'_{rs}(q)} x_\ell \leqslant 1, \quad (r,s) \in P, \tag{32}$$

$$\sum_{\ell \in \mathscr{B}_{rs}(q)} x_\ell = 1, \quad (r,s) \in B, \tag{33}$$

$$x_\ell \in \{0,1\}, \quad \ell \in S(q). \tag{34}$$

We call LB2 the value of the lower bound computed according to expression (15) where UB(q)=min$_\lambda[\sigma''(q,\lambda)]$ and $\sigma''(q,\lambda)$ is the optimal solution cost of RRMP2(q,λ).

5.1. Computation of lower bound LB2

The procedure that we propose for solving problem RRMP2(q,λ) is based on the following observation. Any optimal solution of RRMP2(q,λ) is composed of k $(k \leqslant M)$ submatrices covering the border B and $(M-k)$ submatrices non-covering the border B. Therefore, we can decompose problem RRMP2(q,λ) into two subproblems $S_1(q,\lambda,k)$ and $S_2(q,\lambda,k)$ where $S_1(q,\lambda,k)$ finds among the submatrices of the set $\mathscr{B}(q)$ the k largest cost submatrices covering B (i.e. satisfying constraints (33)) while $S_2(q,\lambda,k)$ finds among the submatrices of the set $\mathscr{B}'(q)$ the $(M-k)$ largest cost submatrices satisfying constraints (32).

Let $\sigma_1(q,\lambda,k)$ and $\sigma_2(q,\lambda,k)$ be the optimal solutions of subproblems $S_1(q,\lambda,k)$ and $S_2(q,\lambda,k)$, respectively; it is quite obvious that the cost $\sigma''(q,\lambda)$ of the optimal solution of RRMP2(q,λ) is given by

$$\sigma''(q,\lambda) = \max_{k \leqslant M}\{\sigma_1(q,\lambda,k) + \sigma_2(q,\lambda,k)\} + \sum_{r=1}^{m}\sum_{s=1}^{n}\lambda_{rs}. \tag{35}$$

Subproblems $S_1(q,\lambda,k)$ and $S_2(q,\lambda,k)$ can be formulated as follows:

$$(S_1(q,\lambda,k)): \quad \sigma_1(q,\lambda,k) = \text{Max} \sum_{\ell \in \mathscr{B}(q)} w'_\ell x_\ell$$

$$\text{s.t.} \sum_{\ell \in \mathscr{B}(q)} x_\ell = k, \tag{36}$$

$$\sum_{\ell \in \mathscr{B}_{rs}(q)} x_\ell = 1, \quad (r,s) \in B, \tag{37}$$

$$x_\ell \in \{0,1\}, \quad \ell \in \mathscr{B}(q) \tag{38}$$

$$(S_2(q,\lambda,k)): \quad \sigma_2(q,\lambda,k) = \text{Max} \sum_{\ell \in \mathscr{B}'(q)} w'_\ell x_\ell$$

$$\text{s.t.} \sum_{\ell \in \mathscr{B}'(q)} x_\ell = M - k, \tag{39}$$

$$\sum_{\ell \in \mathscr{B}'_{rs}(q)} x_\ell \leqslant 1, \quad (r,s) \in P, \tag{40}$$

$$x_\ell \in \{0,1\}, \quad \ell \in \mathscr{B}'(q) \tag{41}$$

5.2. Solving subproblem $S_1(q,\lambda,k)$

Let $G = (X,E)$ be a directed graph where $X = \{1,2,\ldots,|B|\}$ contains one vertex for each element of B (i.e. vertex 1 corresponds to matrix element (1,1), vertex 2

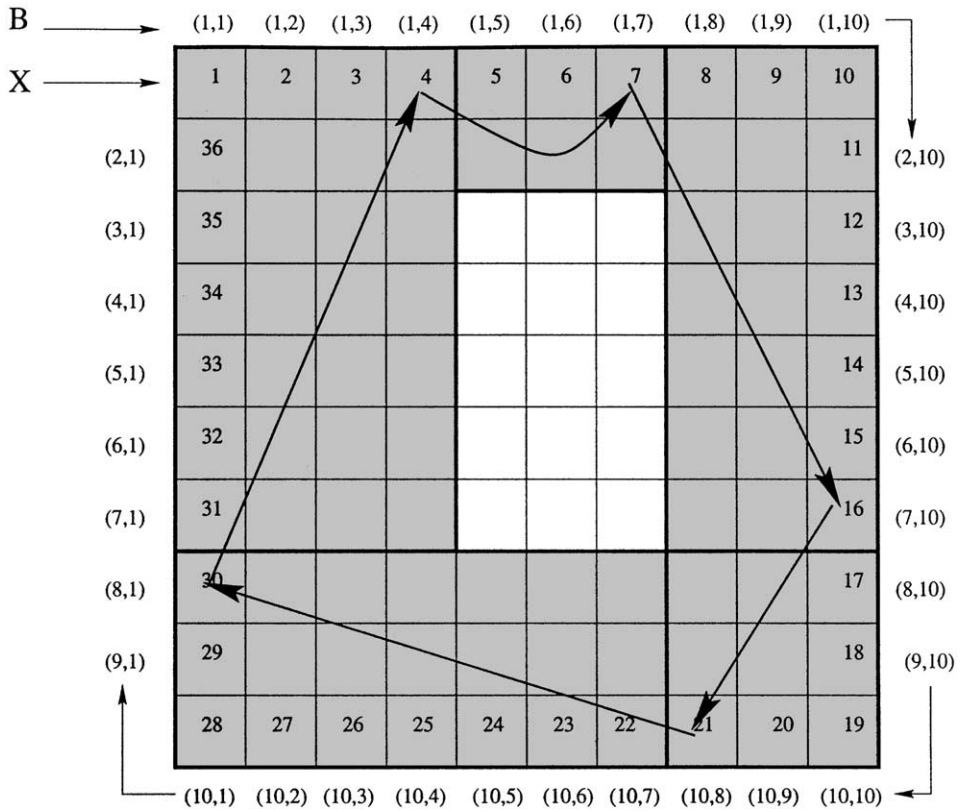

Fig. 5. Example of a circuit in G of cardinality $k = 5$ corresponding to a feasible solution of $S_1(q, \lambda, k)$ for a matrix $A(10 \times 10)$.

corresponds to (1,2), etc. The arc set E contains every pair (i, j) with $i, j = 1, \ldots, |B|$, $i \neq j$, such that:

(i) $i < j$ and $\mathscr{B}(q)$ contains a submatrix ℓ covering the elements of B from position $i + 1$ up to position j;

(ii) $i > j$ and $\mathscr{B}(q)$ contains a submatrix ℓ covering the elements of B from position $i + 1$ to position $|B|$ and from position 1 up to j (see Fig. 5).

With each arc $(i, j) \in E$ is associated a cost

$$c_{i,j} = \mathrm{Max}\{w'_\ell : \ell \in \mathscr{B}(q) \quad \text{s.t. } (r_{i+1}, s_{i+1}), (r_j, s_j) \in R_\ell\}.$$

Every feasible solution of $S_1(q, \lambda, k)$ corresponds to a circuit of cardinality k in G and vice versa. In Fig. 5 it is shown an example of a circuit in G of cardinality corresponding to a feasible solution of $S_1(q, \lambda, k)$ for a matrix $A(10 \times 10)$. The optimal cost $\sigma_1(q, \lambda, k)$ is computed as the cost of the maximum cost circuit of G having cardinality k.

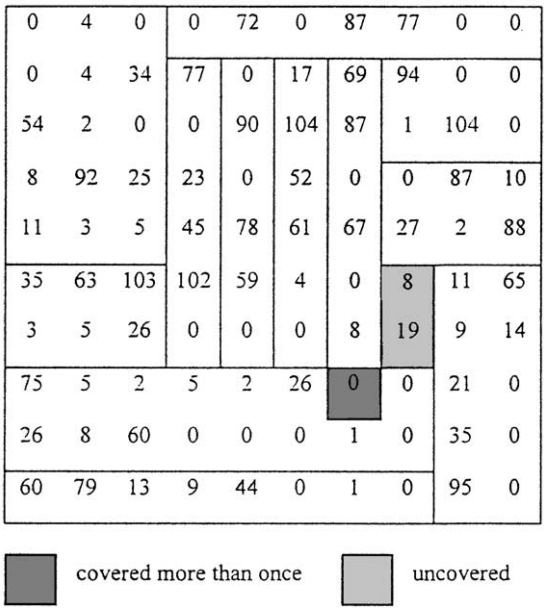

Fig. 6. Example of lower bound LB2 for the problem of Fig. 1.

Notice that every circuit in G of cardinality k contains only one arc (say (i_1, j_1)) with $i_1 > j_1$ (see in Fig .5 the arc $(30,4)$), and $k-1$ arcs $\{(i_2,j_2),\ldots,(i_k,j_k)\}$ with $i_r < j_r$, $r = 2,\ldots,k$ (see in Fig. 5 the arcs $(4,7),(7,16),(16,21),(21,30)$).

The optimal cost $\sigma_1(q,\lambda,k)$ can be computed as follows. Let $E' = \{(i,j)\colon (i,j) \in E$ s.t. $i > j\}$ and $\bar{E} = E \setminus E'$ and let us denote by $f(i,j)$ the cost of the circuit of maximum cost in G of cardinality k containing arc $(i,j) \in E'$. The value of $f(i,j)$ can be computed as $f(i,j) = c_{ij} + g(i,j)$ where $g(i,j)$ is the cost of the path of maximum cost from vertex i to vertex j in the partial graph $\bar{G} = (X,\bar{E})$. Since $\bar{G}$ is an acyclic directed graph, the value of $g(i,j)$ can be computed in polynomial time, see [1]. The optimal solution cost of $S_1(q,\lambda,k)$ is then given by

$$\sigma_1(q,\lambda,k) = \max_{(i,j)\in E'} [f(i,j)].$$

5.3. Solving subproblem $S_2(q,\lambda,k)$

Subproblem $S_2(q,\lambda,k)$ can be solved by inspection in a similar way as RMP1(q,λ). Details of this procedure are given by Mingozzi and Morigi [7].

In Fig. 6 the solution corresponding to LB2 $= 250$ for the matrix given in Fig. 1 is shown. Note that all the border elements are covered, whereas some internal matrix elements remain uncovered.

6. Heuristic method

In this section we propose a heuristic method for finding a feasible MP solution of cost z_{UB} by means of a recursive procedure using guillotine cuts. At the first stage, matrix A is split into two submatrices that are introduced into a stack T. The main recursive step consists of splitting a given submatrix $\ell \in T$, of $\bar{m}(\ell)$ rows and $\bar{n}(\ell)$ columns, by means of either one horizontal or one vertical cut according to a minimum cost criteria. We denote by $\bar{k}(\ell)$ the number of submatrices that, at some earlier stage of the recursion, it has been decided to extract from submatrix ℓ. The cost associated with a horizontal cut in position r is given by the following function $g(r)$:

$$g(r) = \min_{1 \leqslant \alpha \leqslant \bar{k}(\ell)-1} \left[\max \left(\frac{\sum_{i=1}^{r} \sum_{j=1}^{\bar{n}(\ell)} a_{ij}}{\alpha}, \frac{\sum_{i=r+1}^{\bar{n}(\ell)} \sum_{j=1}^{\bar{m}(\ell)} a_{ij}}{\bar{k}(\ell) - \alpha} \right) \right],$$

$$r = 1, \ldots, \bar{m}(\ell - 1), \tag{42}$$

where $g(r)$ is a lower bound to the cost of the submatrix of maximum cost obtained by partitioning submatrix ℓ into $\bar{k}(\ell)$ submatrices using a horizontal cut in position r. Hence the optimal horizontal cut r^* corresponds to $g(r^*) = \min_{1 \leqslant r \leqslant \bar{m}(\ell)} [g(r)]$. We denote by $\eta(r^*)$ the value of α producing the minimum in expression (42).

Similarly, the cost of a vertical cut in column position c is given by the following function $f(c)$:

$$f(c) = \min_{1 \leqslant \beta \leqslant \bar{k}(\ell)-1} \left[\max \left(\frac{\sum_{i=1}^{\bar{m}(\ell)} \sum_{j=1}^{c} a_{ij}}{\beta}, \frac{\sum_{i=1}^{\bar{m}(\ell)} \sum_{j=c+1}^{\bar{n}(\ell)} a_{ij}}{\bar{k}(\ell) - \beta} \right) \right],$$

$$c = 1, \ldots, \bar{n}(\ell - 1). \tag{43}$$

The optimal vertical cut c^* corresponds to $f(c^*) = \min_{1 \leqslant c \leqslant \bar{n}(\ell)} [f(c)]$ and we denote by $\delta(c^*)$ the value of β giving the minimum in expression (43).

At the main recursive step, submatrix l is split into two submatrices ℓ' and ℓ'' either by a guillotine cut at column c^*, if $f(C^*) \leqslant g(r^*)$, or by a guillotine cut at row r^*, if $f(c^*) > g(r^*)$. In the first case we set $\bar{k}(\ell') = \delta(c^*)$ and $\bar{k}(\ell'') = \bar{k}(\ell) - \delta(c^*)$, and in the latter case we set $\bar{k}(\ell') = \eta(r^*)$ and $\bar{k}(\ell'') = \bar{k}(\ell) - \eta(r^*)$. Submatrix ℓ is removed from the stack T and the two submatrices ℓ' and ℓ'' are added to T.

The recursive step is then repeated until M submatrices have been produced (i.e. $|T| = M$).

Fig. 7 shows the solution of cost $z_{UB} = 277$ produced by the heuristic procedure for the problem of Fig. 1.

7. Exact tree search algorithm

We chose to solve problem MP using a binary, depth-first tree search procedure that builds up a complete partitioning of matrix A. The state of a node α of the tree is

0	4	0	0	72	0	87	77	0	0
0	4	34	77	0	17	69	94	0	0
54	2	0	0	90	104	87	1	104	0
8	92	25	23	0	52	0	0	87	10
11	3	5	45	78	61	67	27	2	88
35	63	103	102	59	4	0	8	11	65
3	5	26	0	0	0	8	19	9	14
75	5	2	5	2	26	0	0	21	0
26	8	60	0	0	0	1	0	35	0
60	79	13	9	44	0	1	0	95	0

Fig. 7. Example of heuristic solution for the problem of Fig. 1.

represented by two sets $S0$ and $S1$. The set $S0$ contains the indices of the submatrices that are excluded from the solution (i.e. $x_\ell = 0,\ \forall \ell \in S0$) while $S1$ contains the indices of the submatrices that are fixed in solution (i.e. $x_\ell = 1,\ \forall \ell \in S1$). Both LB1 and LB2 can be easily adapted to cope with the setting of variables $\{x_\ell\}$ for computing a lower bound for the optimal completion of the solution emerging from node α. This involves the setting of $x_\ell = 0,\ \forall \ell \in S0,\ x_\ell = 1,\ \forall \ell \in S1$ and $x_\ell = 0,\ \forall \ell \in S2$ where $S2$ denotes the submatrices that overlap with some submatrix $\ell' \in S1$ (i.e. $S2 = \{\ell: \ell \in S \setminus S0$, s.t. $R_\ell \cap R_{\ell'} \neq \emptyset$ for some $\ell' \in S1\}$). Moreover, in expression (14) we must replace $\sigma(q) = mn$ with $\sigma(q) = mn - \sum_{\ell \in S1} |R_\ell|$.

We use $\bar{S} = S \setminus (S0 \cup S1 \cup S2)$ to denote the submatrices that can belong to the completion of the emerging solution. We backtrack in the tree when the lower bound is greater than or equal to z_{UB} or when there exists an element a_{rs} of matrix A that cannot be covered by any of the submatrices of $\bar{S}$ (i.e. $(r,s) \notin R_\ell, \forall \ell \in \bar{S}$).

We use the following branching rule. Let $\ell^* \in S2$ be a submatrix of maximum cost c_{ℓ^*} less than or equal to the value of the lower bound that contains the highest, then leftmost, element of matrix A not yet covered by any submatrix of the set $S1$. We generate two nodes α' and α'' of the tree search where at α' we set $x_{\ell^*} = 1$ and at α'' we set $x_{\ell^*} = 0$. The algorithm terminates when all its nodes have been explored.

8. Computational results

The algorithm was programmed on Fortran 77 and run on a Pentium with a 200 MHz CPU and 64 Mbytes of memory. To evaluate the performance of our methods with respect to different type of integer square matrices, we generated three different classes

Table 1
Problems of class 1

n	M	LB0	LB1		LB2		$\|S\|$	z_{UB}	z^*	Exact Algorithms			
										A1		A2	
		Value	Value	Time	Value	Time				Time	Nodes	Time	Nodes
10	4	1132	1153	0.0	1153	0.0	63	1153	1153	0.0	0	0.0	0
	8	566	601	0.4	603	9.9	1690	642	603	2.8	16	10.2	2
	16	283	301	0.5	302	1.9	1210	374	304	3.2	24	22.5	6
16	4	3017	3079	0.2	3079	0.0	447	3079	3079	0.2	0	0.0	0
	8	1508	1531	1.8	1557	14.2	3064	1588	1559	38.5	80	75.2	16
	16	754	773	3.8	778	110.7	7849	882	791	1102.0	1270	10 440.0	402
20	4	4808	4858	0.3	4858	0.0	498	4858	4858	0.3	0	0.0	0
	8	2404	2429	0.6	2429	0.1	1098	2429	2429	0.6	0	0.1	0
	16	1202	1216	4.3	1222	131.7	10 766	1253	1240	4825.0	3582	15 753.0	1622
32	4	12 620	12 745	4.4	12 745	0.3	3006	12 745	12 745	4.4	0	0.3	0
	8	6310	6340	13.7	6365	45.4	8872	6388	6388	1017.0	220	1754.0	120

of test problems in order to simulate typical instances of the applications described in Section 1.

Class 1: Uniformly random in the range $[0, 100]$.

Class 2: Thirty percent fixed to zero, 40% uniformly random in the range $[1, 100]$, and 30% in the range $[101, 1000]$.

Class 3: Twenty percent fixed to zero, 20% uniformly random in the range $[1, 10]$, 20% in the range $[11, 100]$, 20% in the range $[101, 1000]$ and 20% in the range $[1001, 10000]$.

Tables 1, 2 and 3 report the computational results of the test problems of classes 1, 2, and 3, respectively. In the tables we give for each problem the size (n) of the square matrix A, the number of partitions (M) to be produced, the value and the computing time of the lower bounds (LB0, LB1, and LB2) at the root node of the tree search, the number of variables ($|S|$) of problem MP, the optimal solution cost (z^*), the value of the upper bound (z_{UB}) produced by the heuristic algorithm of Section 6 and the computing time and the number of nodes of the exact algorithms A1 and A2, where A1 denotes the tree search procedure described in Section 7 using the lower bound LB1, and A2 is the tree search using LB2.

All computing times shown are in seconds. The three tables show that lower bound LB2 is always better than LB1, but LB2 requires more computing time. On average the exact algorithm A2 is computationally more expensive than A1 but it generates a smaller number of nodes in the tree search due to the better quality of lower bound LB2.

We can observe, for the three classes of problems, that the computing time of A1 and A2 increases for increasing values of M due to the fact that the quality of both LB1 and

Table 2
Problems of class 2

n	M	LB0	LB1	LB2			$\|S\|$	z_{UB}	z^*	Exact Algorithms			
										A1		A2	
		Value	Value	Time	Value	Time				Time	Nodes	Times	Nodes
10	4	691	745	0.1	745	0.1	333	745	745	0.6	0	0.0	0
	8	346	367	0.4	371	5.3	1614	394	371	1.4	10	5.3	0
	16	173	190	0.3	190	3.0	1185	209	190	0.6	4	5.1	2
16	4	11 772	11 888	1.6	11 888	4.4	4277	13 533	11 888	1.6	0	4.4	0
	8	5886	6024	8.6	6033	177.3	12 721	7113	6101	112.5	42	925.8	20
	16	2933	3052	5.4	3052	34.6	8479	3550	3110	1240.0	1042	9846.2	368
20	4	15 094	15 472	1.7	15 472	0.6	3042	15 959	15 472	1.7	0	0.6	0
	8	7547	7617	6.7	7699	68.6	10 753	8103	7830	438.6	296	7111.0	264
	16	3774	3784	5.2	3788	180.7	18 783	4126	3908	10 374.0	6074	11 020.0	2132
32	4	48 312	49 057	17.8	49 057	1.4	10 015	49 928	49 057	17.8	0	1.4	0
	8	24 156	24 217	24.2	24 342	134.0	10 954	24 520	24 508	3560.0	568	18 220.0	500

Table 3
Problems of class 3

n	M	LB0	LB1	LB2			$\|S\|$	z_{UB}	z^*	Exact Algorithms			
										A1		A2	
		Value	Value	Time	Value	Time				Time	Nodes	Times	Nodes
10	4	52 010	54 007	0.2	54 007	0.1	200	54 461	54 007	0.2	0	0.1	0
	8	26 005	27 008	0.7	27 412	12.2	1575	29 511	27 701	19.7	58	76.1	14
	16	13 002	15 134	0.6	15 134	3.6	1165	15 420	15 166	61.3	524	635.9	276
16	4	121 769	128 718	1.2	128 718	0.1	1658	130 084	128 718	1.2	0	0.1	0
	8	60 884	61 487	6.6	62 700	132.1	9472	68 235	63 276	305.2	144	793.1	20
	16	30 442	31 677	5.9	31 677	31.0	7707	34 707	33 528	2255.2	1252	7930.2	186
20	4	209 007	214 275	2.7	214 485	0.1	1336	214 485	214 485	5.1	4	0.1	0
	8	104 503	105 175	7.4	105 989	58.8	7663	110 337	107 505	604.7	240	1427.0	106
	16	52 251	52 943	16.8	53 006	188.1	18 560	57 321	54 008	14 007.0	3108	8671.8	242
32	4	468 574	477 082	23.9	477 082	0.6	5985	476 781	477 082	39.4	2	0.7	0
	8	234 287	234 288	6.1	234 288	173.4	19 798	240 442	234 288	7544.1	452	8056.0	238

LB2 deteriorates with M. Furthermore, the results seem to indicate that the performance of algorithms A1 and A2 does not depend on the class of matrix considered in our computational experience.

In conclusion, the exact algorithms A1 and A2 proposed are capable of solving moderately sized MP problems.

References

[1] R.K. Ahuja, T.L. Magnanti, J.B. Orlin, Network Flows, Prentice-Hall, Englewood Cliffs, NJ, 1993.
[2] J.E. Beasley, An exact two-dimensional non guillotine cutting tree search procedure, Oper. Res. 33 (1985) 49–63.
[3] R.I. Becker, I. Lari, M. Lucertini, B. Simeone, Max–min partitioning of grid graphs into connected components, Networks 32 (1998) 115–125.
[4] M.L. Fisher, The lagrangean relaxation method for solving integer Programming problems, Management Sci. 27 (1981) 1–18.
[5] S. Green, Parallel Processing for Computer Graphics, Pitman, London, 1991.
[6] J. De Keyser, D. Roose, Load balancing data parallel programs on distributed memory computers, Parallel Comput. 19 (1993) 1199–1219.
[7] A. Mingozzi, S. Morigi, Min–max partitioning of a matrix with non-guillotine cuts, Technical Report, Department of Math., University of Bologna, 1996.
[8] A. Mingozzi, S. Ricciardelli, M. Spadoni, Partitioning a matrix to minimize the maximum cost, Discrete Appl. Math. 62 (1995) 221–248.
[9] T. Priol, K. Bouatouch, Static load balancing for a parallel ray tracing on a MIMD hypercube, Visual Comput. 5 (1989) 109–119.
[10] N.G. Shivaratri, P. Krueger, M. Singhal, Load distributing for locally distributed systems, IEEE Comput. 25 (1992) 33–44.
[11] R.D. Williams, Performance of dynamic load balancing algorithms for unstructured mesh calculations, Concurrency: Practice Experience 3 (1991) 456–481.
[12] X. Yuan, C. Salisbury, D. Balsara, R. Melhem, A load balancing package on distributed memory systems and its application to particle–particle, particle–particle mesh methods, Parallel Comput. 23 (1997) 1525–1544.